SHR-8S 小型人形机器人
软件说明书

北京森汉科技有限公司

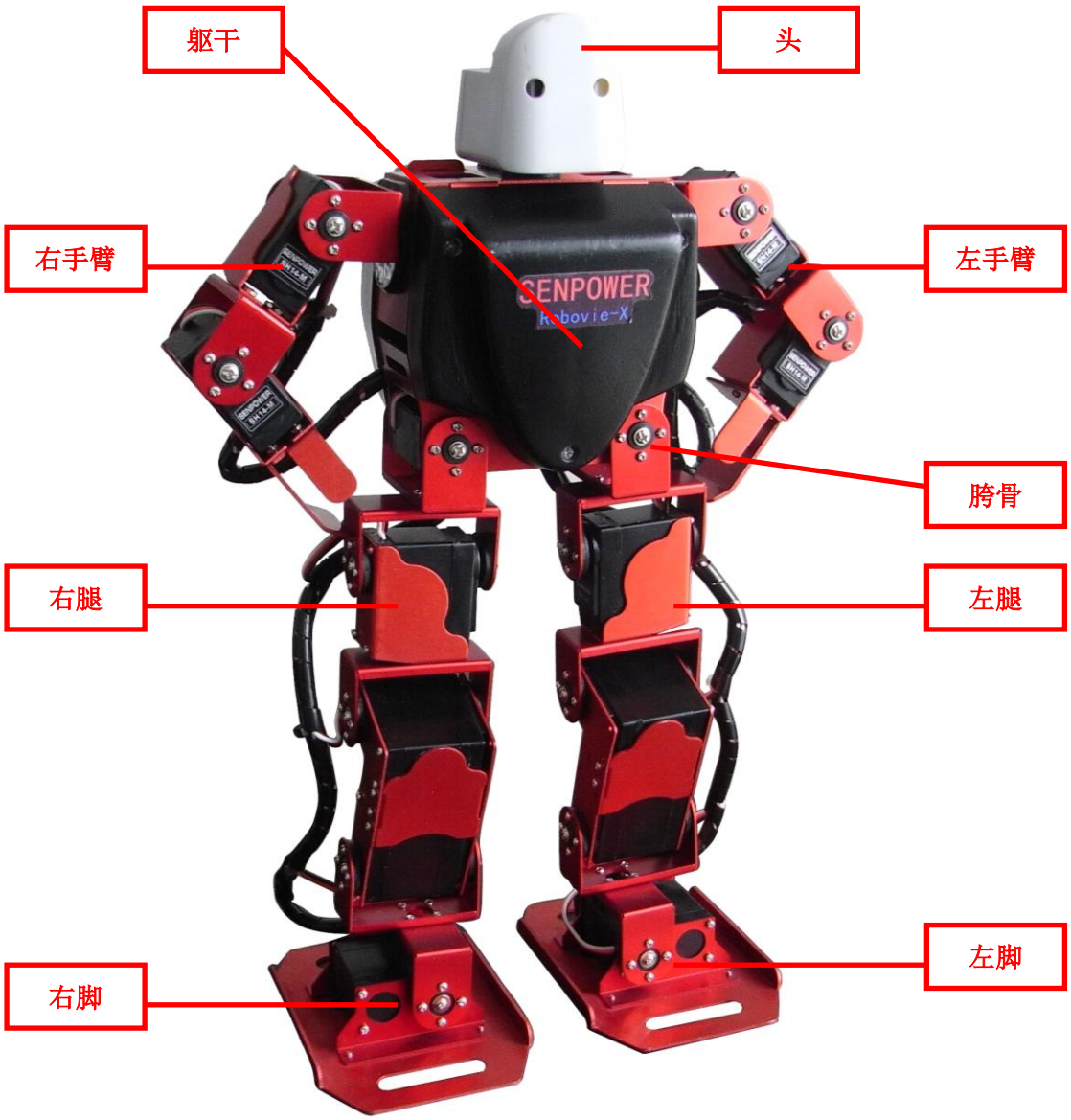
www.senpower.cn

目录

一．SHR-8S 人形机器人机体介绍

（1）关于加藤一郎结构体

基本上，从结构方面看还是比较相似的。这样就对了，早在 1966 年，日本早稻田大学的加藤一郎教授，即国际人形机器人之父就把人形机器人给定型了。头、躯干、四肢的防人结构和被学术界简化并赋予一定数学方程式的数学模型已经注定的现阶段的人形机器人的基本结构。这样做的好处是国际统一与各国之间的技术接轨。我们习惯的称其为“加藤一郎结构体”。如下图分析：



北京森汉公司人形机器人 SHR-8S

（2）制造工艺

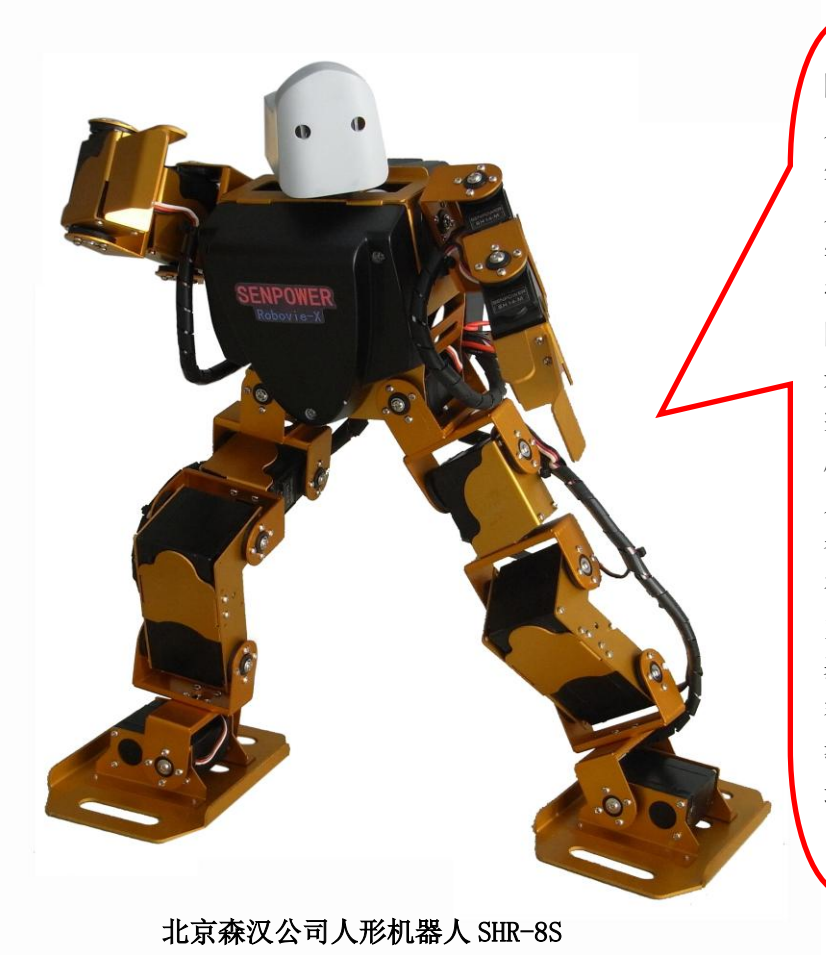
由于目前的国际化生产模式趋于完善，大多厂商的产业链条搭建结构重叠，再加上中国目前处于一个世界超级的外贸加工出口型国家，所以森汉科技生产的小型人形机器人与日本生产的小型人形机器人工艺水平十分接近。

有的日本品牌的小型人形机器人也在中国生产，并且与森汉科技公司的生产线挨得很近。甚至有的零件采用同一条生产线制造，所以共通性较强。也有的日本厂商干脆委托森汉公司代理生产一些核心零件。这样的一个大环境下，致使森汉科技生产的小型人形机器人与日本品牌的小型人形机器人制造工艺十分相似。

实际生产过程中，由模具制造先进行，然后再生产全部的零件。其中包括金属零件和塑料零件。SHR-8S 人形机器人全身共需 19 付模具完成零件生产。

2. 教学小型人形机器人功能介绍

（1）结构教学—人形机器人结构入门



北京森汉公司人形机器人 SHR-8S

除了了解国际标准的人形机器人结构意以外，通过小型人形机器人的实际组装及调试学习，可以直接把学生带进兴趣中来、带进实际中来、带进理论中来，可以实地将理论与实践结合，以最快的速度认知机器人，并且对人形机器人与普通机器人进行概念上的洗礼。为今后的学习与研究过程中打好坚实的基础。而且，该部分内容是参照日本的标准教科书撰写的，与国际最先进的机器人技术国家直接接轨。

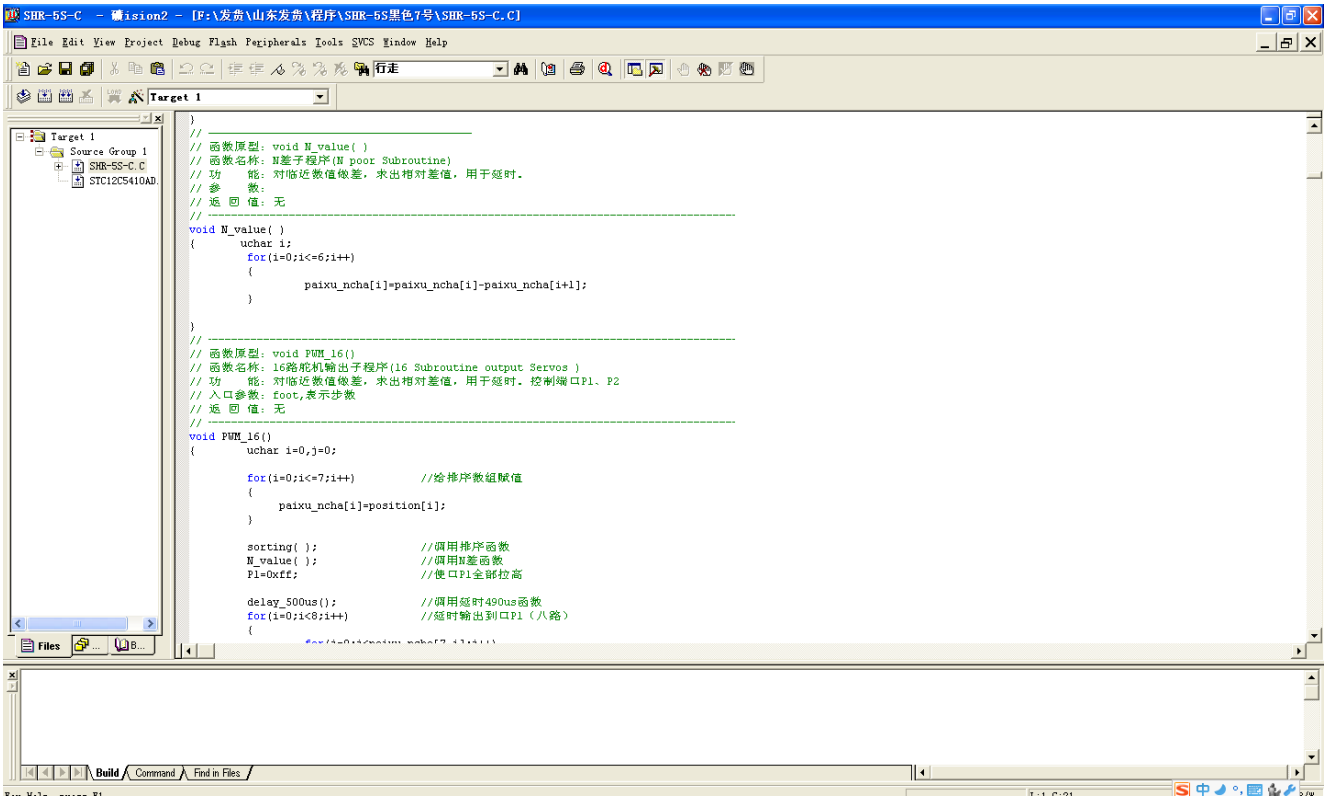
（2）软件教学--人形机器人编程入门

用于编程的机器人软件主要分为 2 类：

1．图形化编程软件，属于应用型软件。使用 IDE 软件开发，主要供广大对计算机知识不了解的人员使用，起科学普及功效。

2．指令级编程软件，属于开发型软件。主要供对计算机知识熟悉的人员使用，起教学研发功效。

3．大型综合性软件，属于大软件工程，目前国际还没有一款完全通用的并且统一的大型通用机器人软件问世，这是一个既可以做普及又可以做研究的综合软件，国际上还处在开发调试阶段。



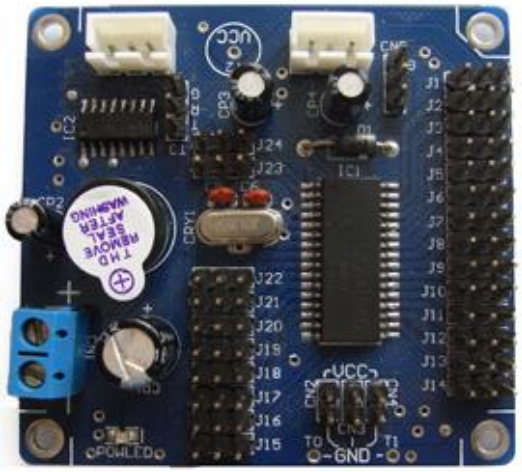
北京森汉公司人形机器人专用 C 语言编程环境

通过这些种针对机器人的编程软件的了解与学习，彻底打破了以往的计算机软件教学的枯燥性与死板性，使学生间接地学习到了相关的计算机知识，并马上联想到实际应用，使计算机知识得以普及并带动后续的机器人知识的教学与普及。

国际上，把计算机软件与机器人编程结合在一起已经不新鲜了，现在就是要尽量的做好普及工作，在普及工作中，人形机器人的无穷乐趣给这项事业带来了新的生机。这就是利用小型人形机器人进行软件教学的功效所在。

二．机器人电路板硬件基础

除了机器人机体本身以外，还有很重要的一个部分就是机器人的控制器系统。森汉科技生产的控制板具有很好的操作性。目前最新市面上流行的 STC 系列单片机许多性能优点，我们的控制板采用 STC12C5412AD 单片机，采用全表贴工艺。对于想学习高速单片机的同学来说，该款单片机是非常全面的，此单片机运行速度快，内部模块丰富。使用这种单片机控制机器人运动，可以使机器人做出更加复杂的动作，同时也可以加入一些具有创意性的设计，这与单片机资源丰富是分不开的。控制板的俯视图如下图所示：



STC12C5412AD 控制板

电路板上面的特殊元件：

EEPROM----片内 12K 的 EEPROM 存储器空间，使机器人程序存储器空间扩充至 12Kbit；

蜂　鸣　器----发出响声，与用户进行信息交互；

红外接受器----接受遥控器的红外信号，与用户进行信息交互；

新增 IO 口----本次的电路板新增至 IO 共有 22 个，其中 17 个用于舵机控制，4 个用于外部采样，数字量输入和模拟量输入；

编程接口----采用国际标准 ISP 串行编程方式；

此控制板设计合理，操作十分简单。简单的操作就可以给芯片编程，而无需插拔芯片。我们为用户开发的 C 语言和汇编语言底层程序，并做成子函数或者子程序。经过简单了解之后，我们就可以编制自己的程序。

1. 人形机器人主板原理图

由于主控板采用 STC12C5412AD 芯片作为主 CPU，那么在板上的其他外围器件要按照 STC12C5412AD 的电气标准设计。该电路板采用 DC7.4V 锂电池供电，经过变压芯片的处理，电压降至 DC5V，稳压。所以，全板芯片采用 TTL 电压设计。CPU 采用外部 16MHZ 晶振，舵机插接端口采用标准 3 芯 2.54 间距插针，用于与舵机的杜邦头相接。

① 有 1 个标准 232 串口插针；

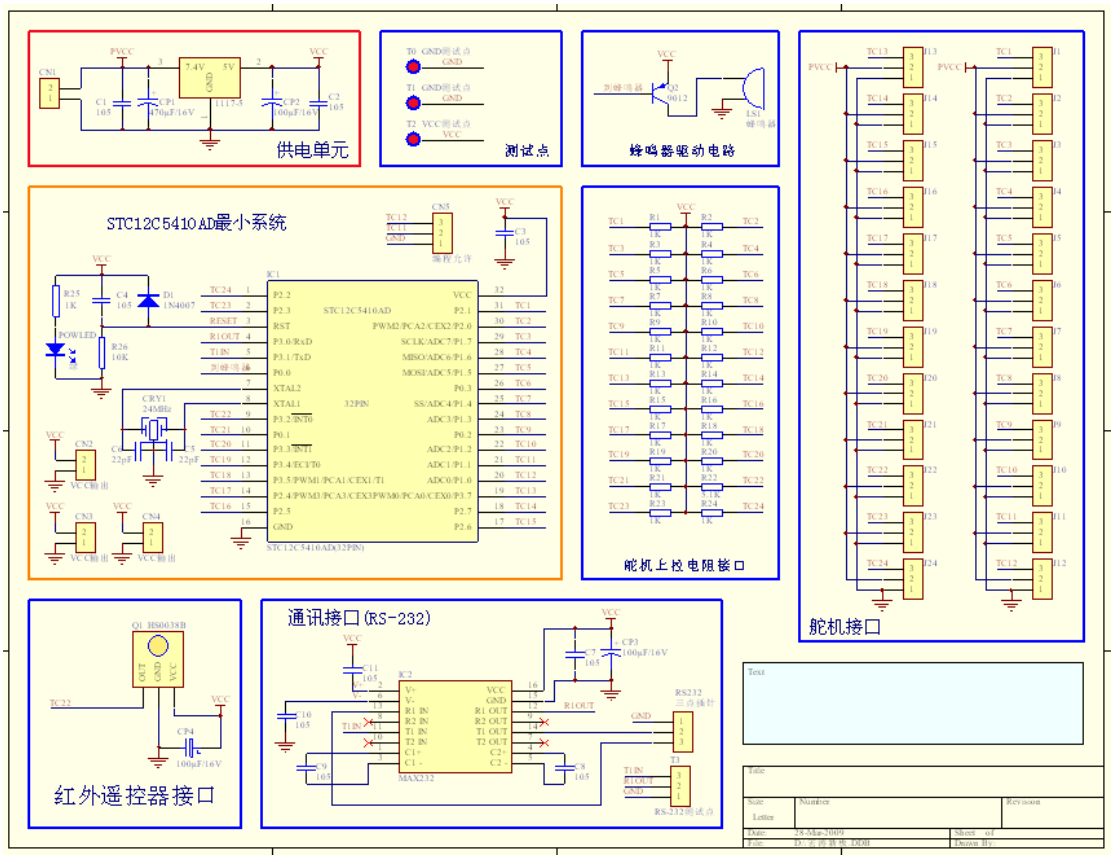
② 有 1 个红外接受头插针，该红外线插针端直接与 CPU 的外部中断相连；

③ 有 1 个带有小功率放大器的蜂鸣器；

④有 22 个舵机插针端口，舵机供电直接转接外部电池 DC7.4V 电压，不用 DC5V 供电；

⑤有 4 个传感器插针，DC5V 供电。分为数字和模拟量输入 2 种形式，与工业标准传感器直接连接就能使用，这样就大大增加了该人形机器人的使用范围；

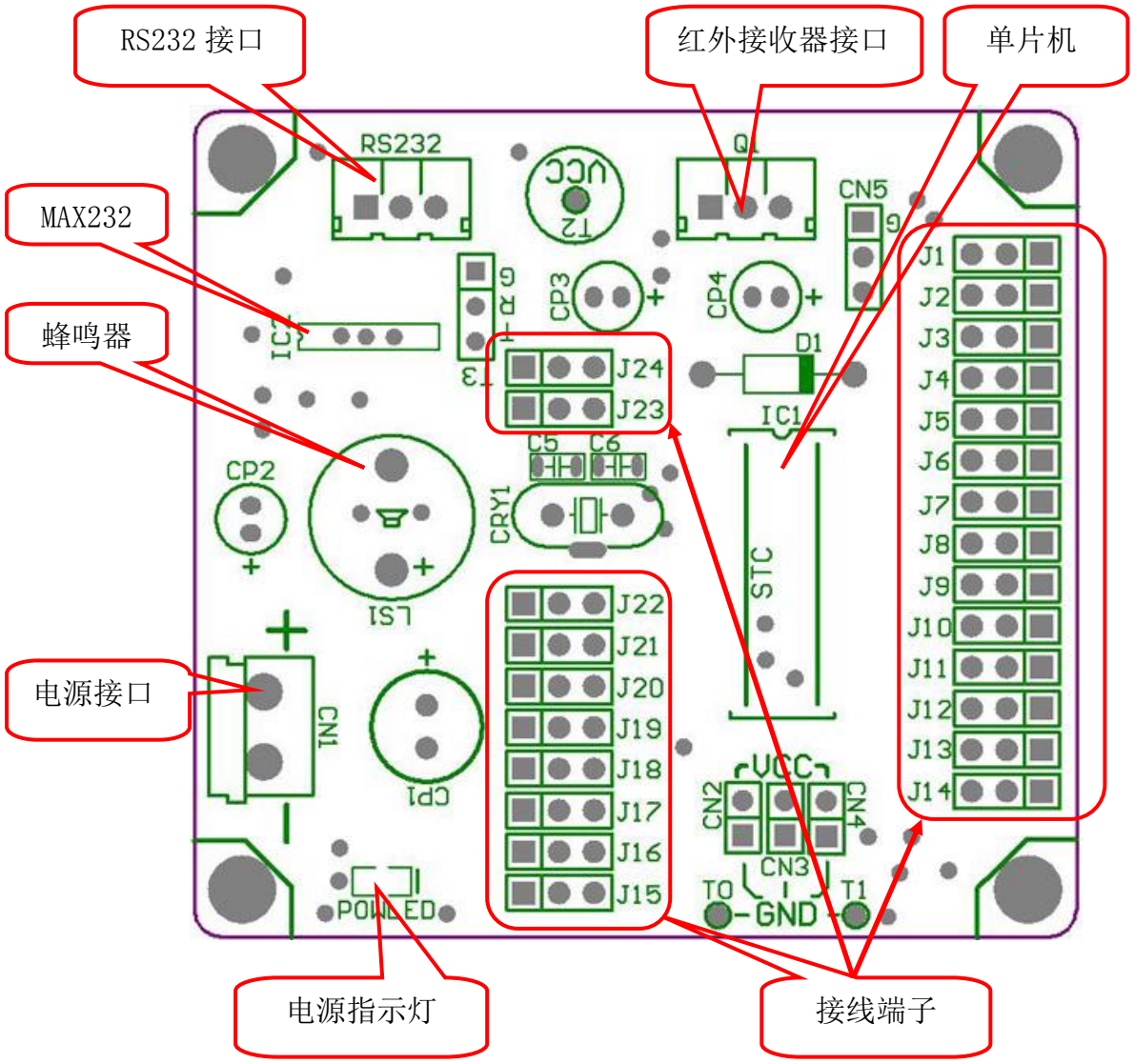
⑥ 有 1 个标准的 DC5V 稳压供电系统；



机器人主控制板原理图

2. STC12C5412AD 控制板 PCB 布局图

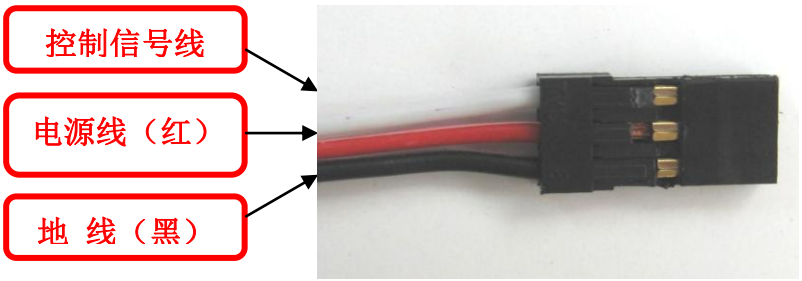
在控制板上有复位按钮和开关。这主要是为了调试程序和上电编程用的。其他部分详见下图说明：



这里需要强调几个注意：

电池的正负一定要看好，一般情况下，红色线是正极，黑色线是负极（GND）。

舵机接线统一遵守标准：靠近单片机的一端是控制信号线，中间是电源（一般是 7.2V 正端），最外面的是地（GND）。这里需要以图片的形式介绍舵机一方的接口定义。如下图所示：

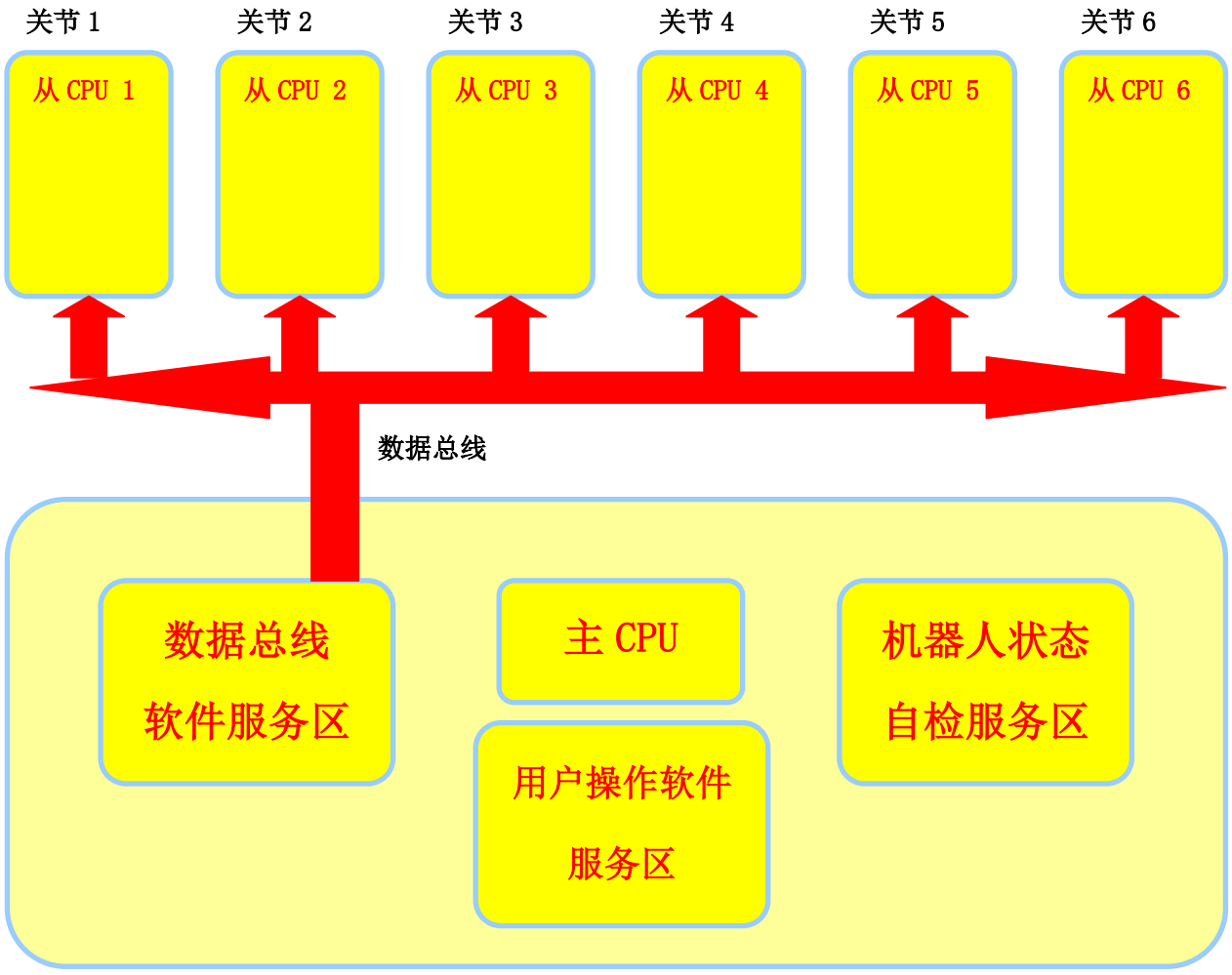


三．全身机电控制系统解析

小型人形机器人因为起步较晚所以显得比较神秘，其实它也很简单，完全属于现代自动控制系统的范畴。我们本次采用主从控制结构。每一个伺服电机里面自带有 CPU 控制板。

1．系统构成与控制方式

跟大多数数控设备一样，小型人形机器人也采用主从式的伺服结构控制方式。即：负责用户交互软件的为主 CPU 即 STC125410AD 芯片。其他的具体执行关节位置及速度命令的为从 CPU，从 CPU 用来控制伺服电机。由于机器人的关节较多，所以整个属于多轴联动系统。程序里面要用到数据通讯、位置跟踪、联动与数值插补等技术。

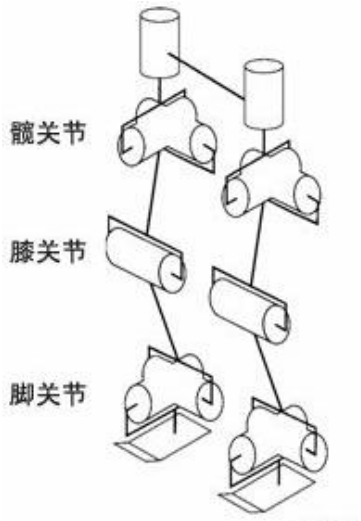


小型人形机器人专用多轴伺服系统

2. 全身机械结构原理

人类在研究人体结构之前花费了大量的时间去研究昆虫，哺乳动物的腿部移动，甚至登山运动员在爬山时的腿部运动方式。这些研究帮助我们更好的了解在行走过程中发生的一切，特别是关节处的运动。比如，我们在行走的时候会移动我们的重心，并且前后摆动双手来平衡我们的身体。这些构成了人形机器人行走的基础方式。

人形机器人和人类一样，有髋关节，膝关节和足关节。机器人中的关节一般用“自由度”来表示。一个自由度表示一个运动可以或者向上，或者向下，或者向右，或者向左。分散在身体的不同部位，所以骨骼结构因此而生。



一般的，人形机器人身上装有两个传感器能辅助它水平行走，它们是加速度传感器和陀螺传感器。它们主要用来让机器人知道身体目前前进的速度以及和地面所成的角度，并依次计算出平衡身体所需要调节量。这两个传感器起的作用和我们人类内耳相同。要进行平衡的调节，机器人还必须要有相应的关节传感器和 6 轴的力传感器，来感知肢体角度和受力情况。

机器人的行走中最重要的部分就是它的调节能力。所以需要检测在行走中产生的惯性力。当机器人行走时，它将受到由地球引力，以及加速或减速行进所引起的惯性力的影响。这些力的总和被称之为总惯性力。当机器人的脚接触地面时，它将受到来自地面反作用力的影响，这个力称之为地面反作用力。所有这些力都必须要被平衡掉，而机器人的控制目标就是要找到一个姿势能够平衡掉所有的力。这称做"zero moment point" (ZMP)。当机器人保持最佳平衡状态的情况下行走时，轴向目标总惯性力与实际地面反作用力相等。相应地，目标 ZMP 与地面反作用力的中心点也重合。当机器人行走在不平坦的地面时，轴向目标总惯性力与实际的地面反作用力将会错位，因而会失去平衡，产生造成跌倒的力。跌倒力的大小与目标 ZMP 和地面反作用力中心点的错位程度相对应。简而言之，目标 ZMP 和地面反作用力中心点的错位是造成失去平衡的主要原因。假若机器人失去平衡有可能跌倒时，下述三个控制系统将起作用，以防止跌倒，并保持继续行走状态。

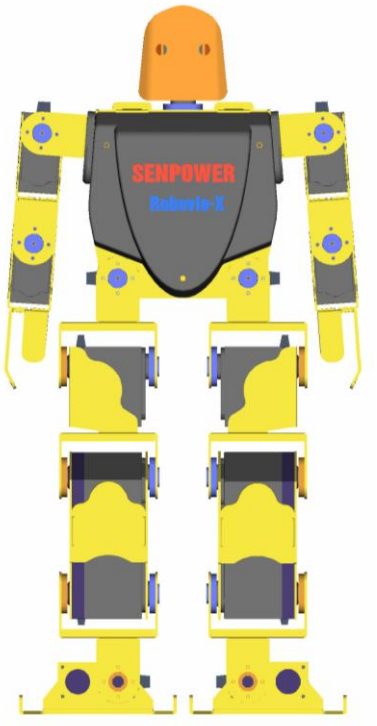
（1）地面反作用力控制：脚底要能够适应地面的不平整，同时还要能稳定的站住。

（2）目标 ZMP 控制：当由于种种原因造成机器人无法站立，并开始倾倒的时候，需要控制他的上肢反方向运动来控制即将产生的摔跤，同时还要加快步速来平衡身体。

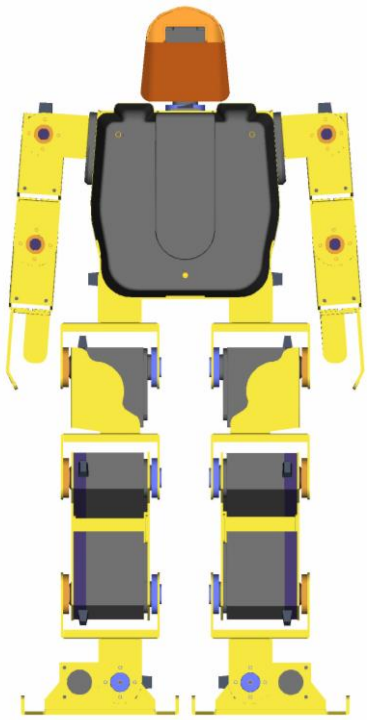
（3）落脚点控制：当目标 ZMP 控制被激活的时候，机器人需要调节每步的间距来满足当时身体的位置，速度和步长之间的关系。

四．SHR-8S 人形机器人机体参数

SHR-8S 人形机器人是 17 自由度小型人形机器人。全身包括 17 个伺服电机（舵机）。每个舵机扭矩为 14kg.cm，全部采用金属齿轮传动。

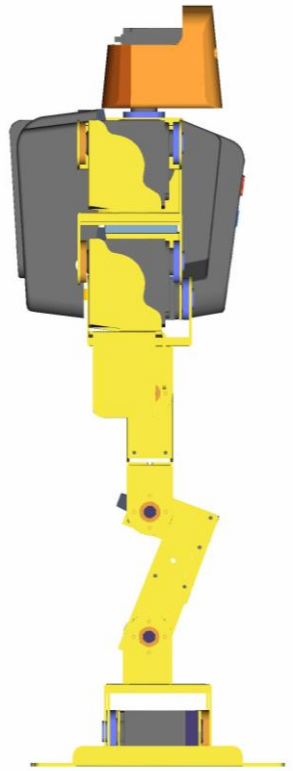


正视图

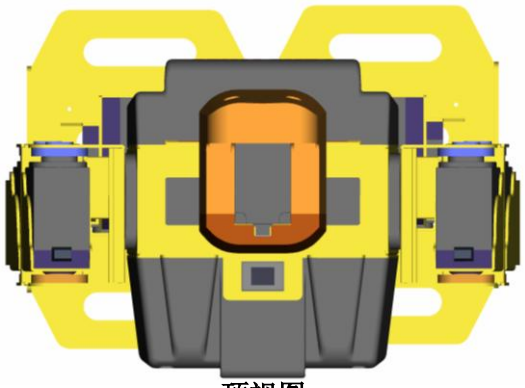


背视图

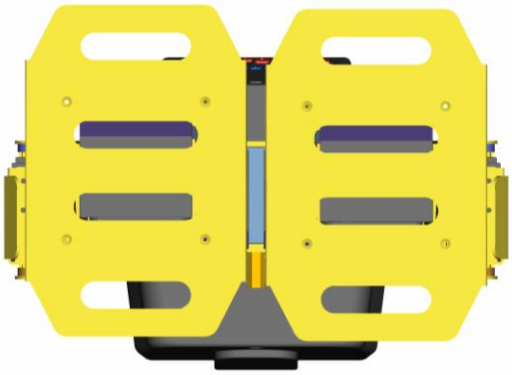
身高：360mm；
肩宽：165mm；
胸厚：115mm；
质量：1.5kg；



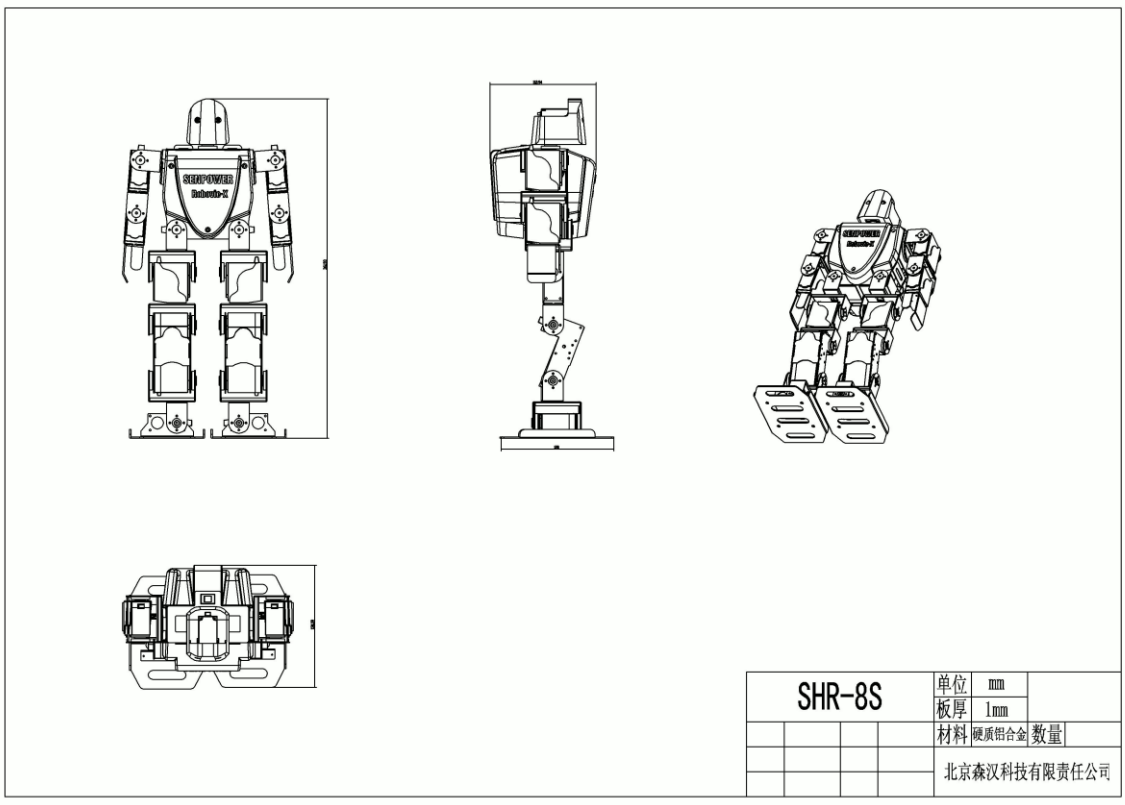
侧视图



顶视图



底视图



五．舵机工作原理

1．PWM 信号的定义

市面上有上千种舵机，有上百家厂商生产舵机，但是它们的 PWM 信号协议都是一样的。PWM 信号为脉宽调制信号，其数据量存在于它的上升沿与下降沿之间的时间宽度，宽度越大表示数据越大，反之越小。我们目前使用的舵机主要依赖于模型行业的标准协议，随着人形机器人行业的渐渐独立，舵机也在不断升级。有些厂商已经推出全新的舵机协议和全新的舵机外壳形状，这些舵机只能应用于机器人行业，已经不能够应用于传统的模型上面了。

目前，北京森汉使用的 SH14-M 舵机采用传统的 PWM 协议。该款舵机已经批量生产，扭矩达到 14kg.cm，旋转角度达到 185 度。该舵机控制方式采用传统 PWM 格式。

由反馈参量的性质决定舵机的性质。SH14-M 舵机是将电位器电压值进行 AD 转换，变成数字量与来自单片机的 PWM 信号转换成的数字量进行数字计算，所以属于数字性质的反馈。所以称其为数字舵机。另外，在该舵机的软件中加入了锁存算法，致使其对 PWM 信号的要求较低，这样可以明显减少主控 CPU 的实时计算量。

（1） 采用固定协议，不用随时接收指令，为 CPU 腾出大量时间干其他事情；

（2） 可以位置自锁、位置跟踪，这方面超越了普通的步进电机；

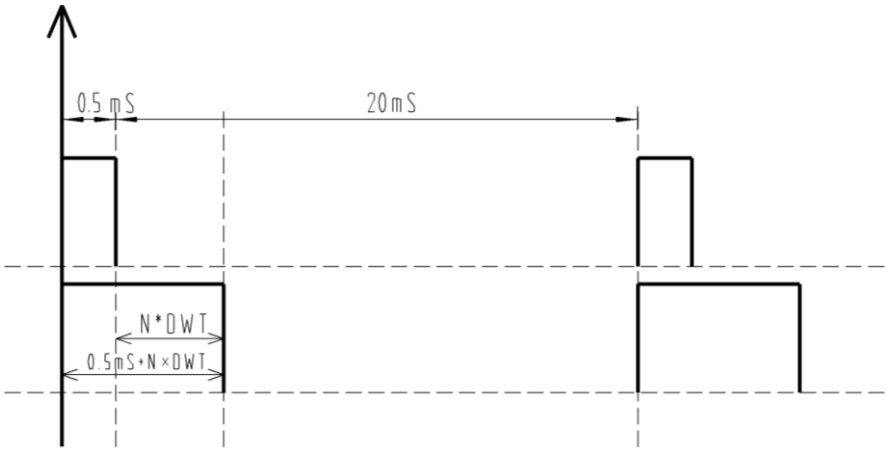


图 1-1

其 PWM 格式注意的几个要点：

（1） 上升沿最少为 0.5mS，为 0.5mS---2.5mS 之间；

（2） SH14-M 数字舵机对 PWM 信号的下降沿时间没要求，我们可以利用这点好处对机器人的舵机进行任意调速；

2. PWM 信号控制精度制定

我们本次采用的是 8 位 AVR 高速 STC12C5412AD 芯片，其数据分辨率为 256，那么经过舵机极限参数实验，应该将其划分为 250 份，省略 5 份。

那么：0.5mS---2.5mS 的宽度为 2mS = 2000uS；

2000uS÷250 = 8uS；

则：**PWM 的控制精度为 8us（一般的，8 位机控制舵机时，时间控制精度都设定成 8us）**

我们以 8uS 为单位递增控制舵机转动与定位。

舵机可以转动 185 度，我们取 180 度，省略边沿的 5 度。

那么：180 度÷250=0.72 度；

则：**舵机的控制精度为 0.72 度；**

1 DWT = 8uS ; 250DWT = 2mS

时基寄存器内的数值为：01 ---- 250。

共 180 度，分为 250 个位置，每个位置叫 1DWT。

则：180÷250 = 0.72 度 / DWT

PWM 上升沿函数： 0.5mS + N×DWT

0uS ≤ N×DWT ≤ 2mS

0.5mS ≤ 0.5 mS + N×DWT ≤ 2.5mS

六．24 路并行积分法控制舵机

所谓的并行积分法控制舵机，就是指相对于每个舵机都是由积分法进行角度转动控制，而且将 8-24 个舵机同时进行积分就构成了并行积分。这种方法的好处是对单个舵机而言，属于最高精度控制，每一个转动微分元的精度都能够达到单片机所能保证的最高值。而且积分的时间可以通过延时程序进行控制，达到柔滑的调速效果。另外，由于采用并行算法，所以在一个标准 PWM 周期内，也就是说在 2.5ms 内可以完成 8-24 个舵机的控制。这对于关节很多的数控系统至关重要，尤其是人形机器人。

1. 单个舵机的转角特性

（1）当其未转到目标位置时，将全速向目标位置转动。

（2）当其到达目标位置时，将自动保持该位置。

所以对于数字舵机而言，PWM 信号提供的是目标位置，跟踪运动要靠舵机本身。

（1）SH14-M 舵机的角度控制方法

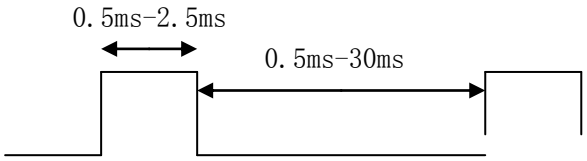
舵机的转角达到 180 度，由于采用 8 为 CPU 控制，所以控制精度最大为 255 份。目前经过实际测试和规划，分了 250 份,省略 5 份。

将 0—180 分为 250 份，每份 0.72 度。

控制所需的 PWM 宽度为 0.5ms—2.5ms，宽度 2ms。

2ms÷250=8us；

所以得出：**PWM 信号 = 1 度/8us**；



舵机角度= 0.72 度×N

PWM = 0.5mS + N×DWT；（DWT=8us）

角度	0	45	90	135	180
N	0	62	125	187	250
PWM 粗算	0.5ms	1ms	1.5ms	2ms	2.5ms

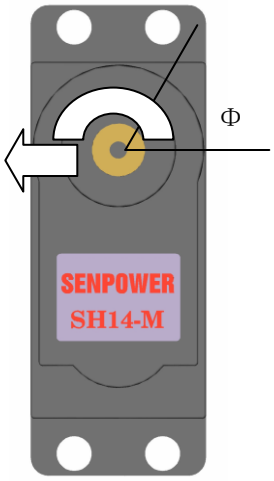
角度	0	44.64	90	134.64	180
N	0	62	125	187	250
PWM 精算	500us	996us	1500us	1996us	2500us

（2）SH14-M 舵机的方向制定

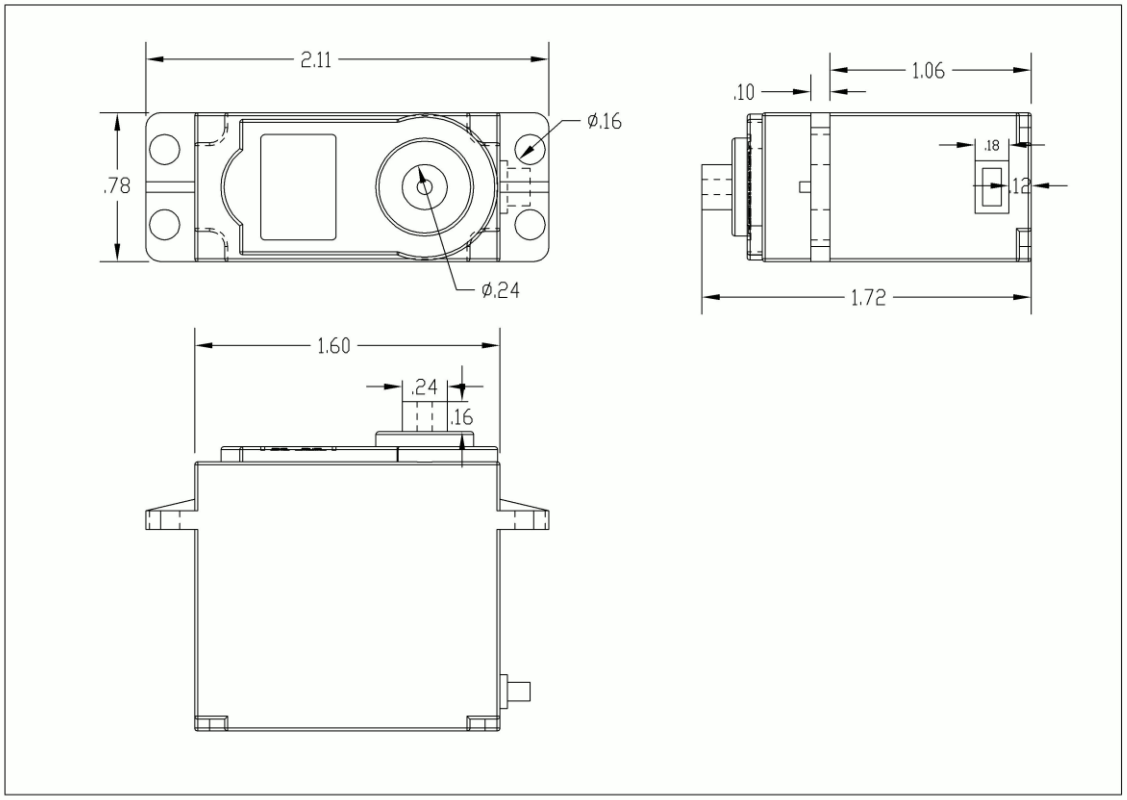
舵机的转动方向为:
逆时针为正转

Φ对应 N 值
N= 0 ， Φ=0 度
N=250， Φ=180 度

1 ≤ N ≤ 250



运动时可以外接较大的转动负载，舵机输出扭矩较大，而且抗抖动性很好，电位器的线性度较高，达到极限位置时也不会偏离目标。



上图采用英制单位

2. 单舵机 DWA 的定义

将 180 度的转角分为 250 个平均小份。
则：每小份为 0.72 度。

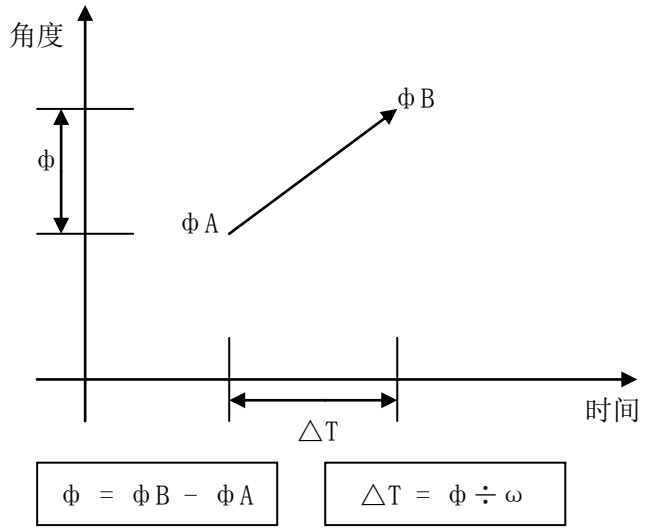
定义如下：DWA = 0.72 度

由于： ω = 0.2 秒/60 度
则：运行 1 DWA 所需时间为：0.72 度÷60 度 /0.2 秒= 2.4 mS；

3. 单舵机 DWT 的定义

舵机电路支持的 PWM 信号为 0.5mS—2.5mS，总间隔为 2mS。
若分为 250 小份，则 2mS÷250 = 0.008 mS = 8uS

定义如下：DWT = 8uS



那么 1 DWA（0.72 度）对应的△T 为：0.72 度÷60 度/0.2 秒 =2.4mS。

则：①舵机转动 1DWA----PWM 通讯时间单位为 8uS；
②舵机转动 1DWA----PWM 通讯总时间为 2.5mS；
③舵机转动 1DWA----实际转动过程时间为 2.4mS；
根据上面②和③可以看出，国际上通用 PWM 协议中的 2.5mS 的来历，源自舵机本身的角速度。

4. 24 路舵机的并行控制

（1）控制要求

要求同时发出 24 个国际标准 PWM 信号，并且在一个周期内完成。
效果：可以轻松扩充至 32 个舵机；信号传递速率高，舵机的跟随特性好；
我们以 24 路为 1 组同时进行积分，24 个舵机在跟随过程中完全复合积分算法；并且实现 24 路同时调速；

（2）注意事项

从 24 个端口，PA、PB 到 PC，单 DWT 循环的最小时间只有 8us，所以串行运算是不行的，那么就采用并行运算。

目前采用的并行算法是 PA.0—PA.7 为一个基本单位，8 位一并。3 个端口 PA、PB、PC 采用共用体函数同时发出脉冲。实际案例：PA 口的 8 个位置个不相同；

端口	N 寄存器	目标位置（度）	N 数值（整数）	PWM 宽度（ms）
PA .0	Loc [0]	0	0	0.500
PA .1	Loc [1]	0.72	1	0.508
PA .2	Loc [2]	45	62.5	1.000
PA .3	Loc [3]	50	67.6	1.041
PA .4	Loc [4]	60	81.1	1.148
PA .5	Loc [5]	90	125	1.500
PA .6	Loc [6]	135	187.5	2.000
PA .7	Loc [7]	180	250	2.500

注意：N 为整数，依照上表看出，由于整数原因，舵机定位不能实现的有 45 度、50 度、60 度、135 度等。上表中红色的数据是实际不存在的。

端口	N 寄存器	目标位置（度）	N 数值（整数）	PWM 宽度（ms）
PA .0	Loc [0]	0	0	0.500
PA .1	Loc [1]	0.72	1	0.508
PA .2	Loc [2]	44.64	62	0.996
PA .3	Loc [3]	48.96	68	1.044
PA .4	Loc [4]	58.32	81	1.148
PA .5	Loc [5]	90	125	1.500
PA .6	Loc [6]	135.36	188	2.004
PA .7	Loc [7]	180	250	2.500

注意：这个表中的数据才是真实存在并可利用 PWM 协议执行的。

（3）并行 PWM 信号发生算法解析

我们预计将整个周期控制在 2.5ms 内；
PA、PB、PC 口的 24 个端在不同时间产生下降沿，并且分为 3 组；
那么由上例如：我们的 PA.5 口，他的 N 为 85
那么就需要它在 85 个 DWT 后产生下降沿，时间为（85×8us=680us）。
发现 2 个关键参数：①时间参数 N=85；
②逻辑参数 PA.5=（#0DFH）（#0DFH）（#0DFH）（3 位操作“&”关系）；
逻辑参数的定义：采用&指令，操作 PA、PB、PC 口。

&端口逻辑参数表

	PA 端口字节	PB 端口字节	PC 端口字节	备注
PA .0	# FEH	# FFH	# FFH	PA 口操作 PB、PC 口不操作
PA .1	# FDH	# FFH	# FFH	
PA .2	# FBH	# FFH	# FFH	
PA .3	# F7H	# FFH	# FFH	
PA .4	# EFH	# FFH	# FFH	
PA .5	# DFH	# FFH	# FFH	
PA .6	# BFH	# FFH	# FFH	
PA .7	# 7FH	# FFH	# FFH	
PB .0	# FFH	# FEH	# FFH	PB 口操作 PA、PC 口不操作
PB .1	# FFH	# FDH	# FFH	
PB .2	# FFH	# FBH	# FFH	
PB .3	# FFH	# F7H	# FFH	
PB .4	# FFH	# EFH	# FFH	
PB .5	# FFH	# DFH	# FFH	
PB .6	# FFH	# BFH	# FFH	
PB .7	# FFH	# 7FH	# FFH	
PC .0	# FFH	# FFH	# FEH	PC 口操作 PA、PB 口不操作
PC .1	# FFH	# FFH	# FDH	
PC .2	# FFH	# FFH	# FBH	
PC .3	# FFH	# FFH	# F7H	
PC .4	# FFH	# FFH	# EFH	
PC .5	# FFH	# FFH	# DFH	
PC .6	# FFH	# FFH	# BFH	
PC .7	# FFH	# FFH	# 7FH	

以上的逻辑操作数，可以构成一个 24 行 3 列的数组。然后以行为单位排序。

单字节&端口逻辑参数表

PA、PB、PC	PM.7	PM.6	PM.5	PM.4	PM.3	PM.2	PM.1	PM.0	备注
PM.0= # FEH	1	1	1	1	1	1	1	0	3 个端口的 24 个编号整合成3组
PM.1= # FDH	1	1	1	1	1	1	0	1	
PM.2= # FBH	1	1	1	1	1	0	1	1	
PM.3= # F7H	1	1	1	1	0	1	1	1	
PM.4= # EFH	1	1	1	0	1	1	1	1	
PM.5= # DFH	1	1	0	1	1	1	1	1	
PM.6= # BFH	1	0	1	1	1	1	1	1	
PM.7= # 7FH	0	1	1	1	1	1	1	1	

例如：将 PA.5 口产生下降沿，就将#0DFH、#0FFH、#0FFH 去“&” PA、PB、PC 口。
逻辑“&”指令，冯“0”得“0”，不影响其他位。
具体的程序操作如下：

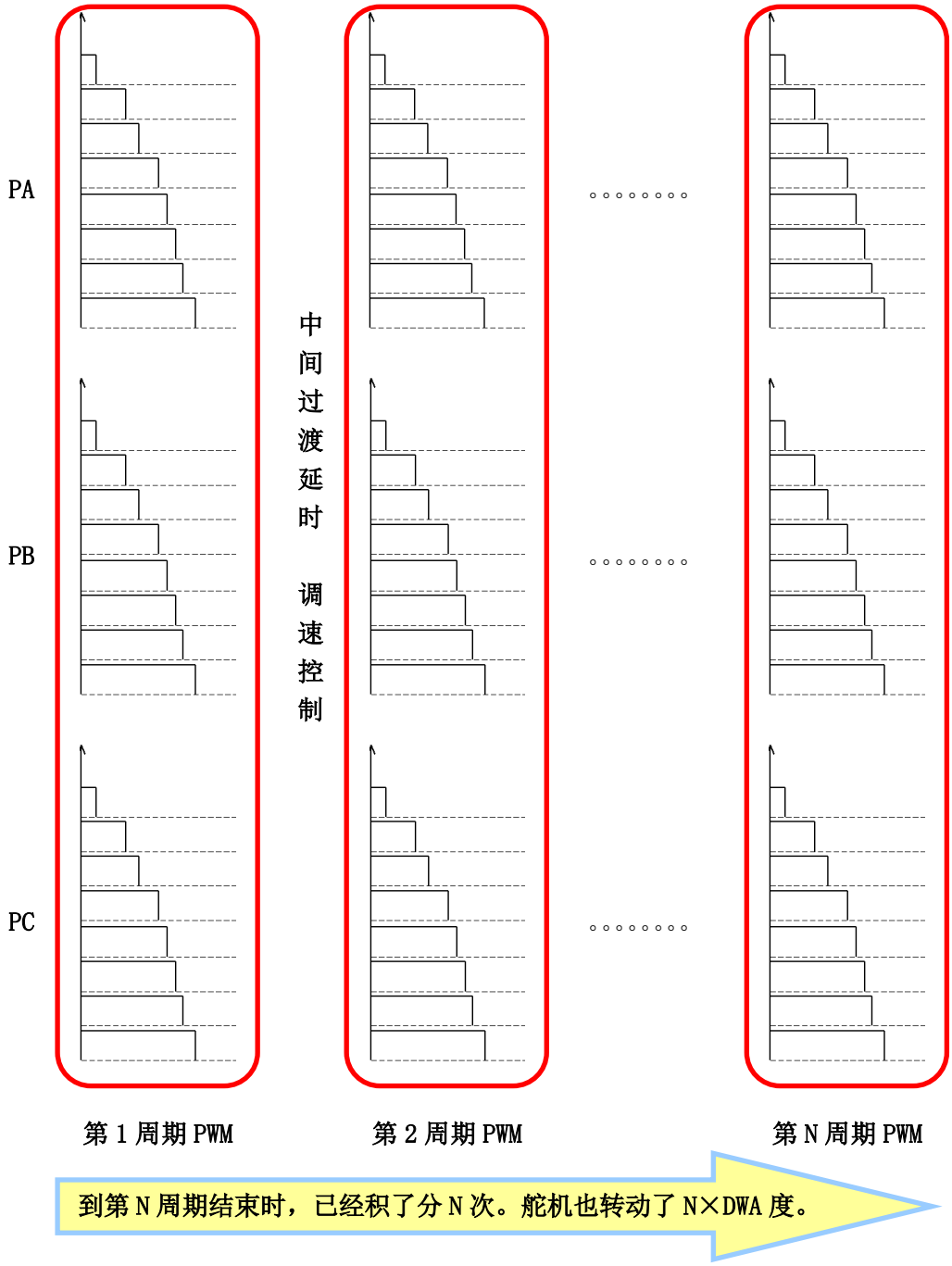
① 开 2.5ms 定时中断。为了后面的集体舞动作，否则不用开中断；
② 取出 24 个端口（PA.0-PA.7）、（PB.0-PB.7）、（PC.0-PC.7）的位置值，也就是 24 个变化量；并赋予相应的 3 字节端逻辑参数；
③ 将这 24 个量按照规律排列，相应端的逻辑参数值也跟随排列；
④ 将 24 个舵机位置值之间求差；
⑤ 得出 delt1，延时 delt1×DWT，& 相应的逻辑参数；
得出 delt2，延时 delt2×DWT，& 相应的逻辑参数；
。。
得出 delt9，延时 delt9×DWT，& 相应的逻辑参数；
得出 delt10，延时 delt10×DWT，& 相应的逻辑参数；
。。
得出 delt17，延时 delt17×DWT，& 相应的逻辑参数；
得出 delt18，延时 delt18×DWT，& 相应的逻辑参数；
。。。
得出 delt24，延时 delt8×DWT，& 相应的逻辑参数；
⑥ 24 个端的下降沿全部产生完毕，等待单周期 2.5ms 中断的到来；
⑦ 中断到来后，清理中断标志，然后结束该程序。

注意事项： 当进行逐个排序延时的过程中，CPU 要取出 3 组 delt1 delt2...delt24。利用 C 语言中的共用体后，一样在 2.5ms 内完成 3 组 8 路共 24 路的排序以及求差工作。

5. 多路并行积分法解析

（1）如何积分

上面讲述了并行法控制舵机，那只不过是在一个标准 PWM 周期（2.5ms）内，将 24 个 PWM 信号发送个 24 个舵机的单步操作。要使人形机器人真正连续的动起来，还需要连续地、不停地发 PWM 脉冲。那么，下面所述的积分法就是连续大量地发 PWM 的一种经典方法。



（2）如何调速

令：整个积分过程的总时间为 T；
积分次数为 N；
每个基本延时为 t；
总转动角度为Φ；

则：$T = N\times 2.5ms + (t_1 + t_2 + t_3 + \cdots\cdots + t_N)$。

总转动角度为：N×0.72 度。

总平均速度为：$\Phi/T = (N\times 0.72\text{度}) / (N\times 2.5ms + t_1 + t_2 + t_3 + \cdots\cdots + t_N)$。

除了能够控制整体速度以外，还能够通过每个微元的积分环节进行高精度的速度控制。

如果把中间过渡延时做成函数形式，那么就可以进行精确地调速控制。

即：t_1、t_2、t_3 。。。。。t_N 是个函数
$t_N = f(N)$
f 是以 N 为自变量的函数

调速方法：我们通过编写各种各样的 $f(N)$ 函数，对多个舵机进行总调速控制。

例如：加速运动---- $f(N)$ 为递减函数；
减速运动---- $f(N)$ 为递增函数；
匀速运动---- $f(N)$ 为常数；

一般地，我们在进行舵机调速时都分为 3 个阶段。加速阶段、匀速阶段、减速阶段。
所以，在后面讲述的关于人形机器人的行走及全身运动时就可以大量应用 $f(N)$ 函数进行平滑的速度控制。

（3）中间过渡延时系数

如上面的调速过程中，调用 bingxing_24()程序后，依靠变化的过度延时程序拖延下次位置数据发送的时间，并将这种规律写入 f（N）函数。

令：过渡延时系数为 Delt

则：根据舵机的经验参数得出，5ms ≤过渡延时时间≤ 25ms ；△过渡延时=20ms；

若：以 0.5 ms 为 1 个时间单位，

则：△Delt= 20ms÷0.5ms = 40

则：1≤△Delt≤40

在进行系统调速控制时，改变系统速度有 2 个方法：

（1）舵机控制分辨率不变，改变下一个目标位置的发出时间，使 delt 变化；

（2）Delt 不变（即每个控制周期均保持 20ms 左右），改变系统的分辨率；

由于好的数控系统，忌讳变更分辨率。我们每时每刻希望得到系统的最高分辨率，所以我们采用改变 Delt 数的方法，控制舵机速度。

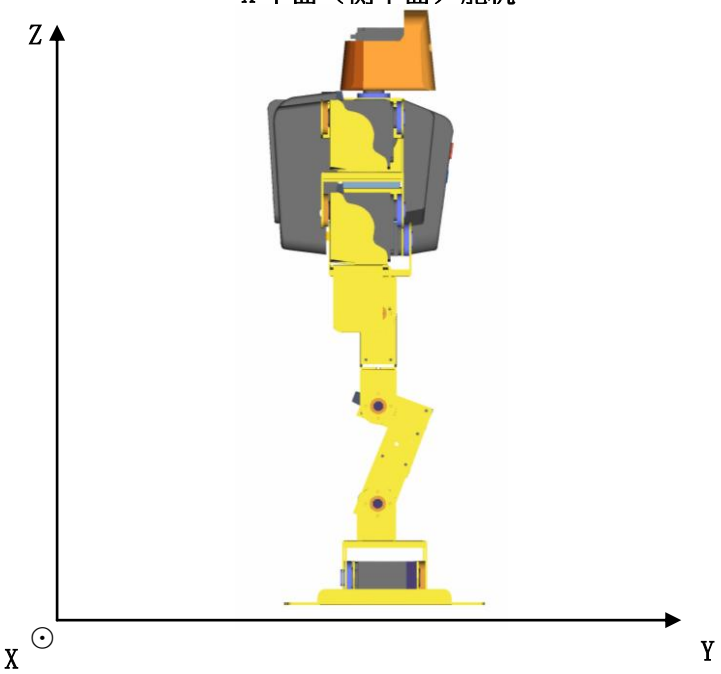
延时系数 Delt 十分重要，在后面的速度与加速度的控制算法中会针对每种运动方式具体给出 Delt 的函数，而这些也是微分元函数表达式的重要组成部分。

七．机器人控制端口定义

由于机器人的机械体与计算机系统相接，所以肯定需要定端口协议。实际上，按照加藤一郎的结构划分，人形机器人分为 3 个坐标平面。

1．平面定义

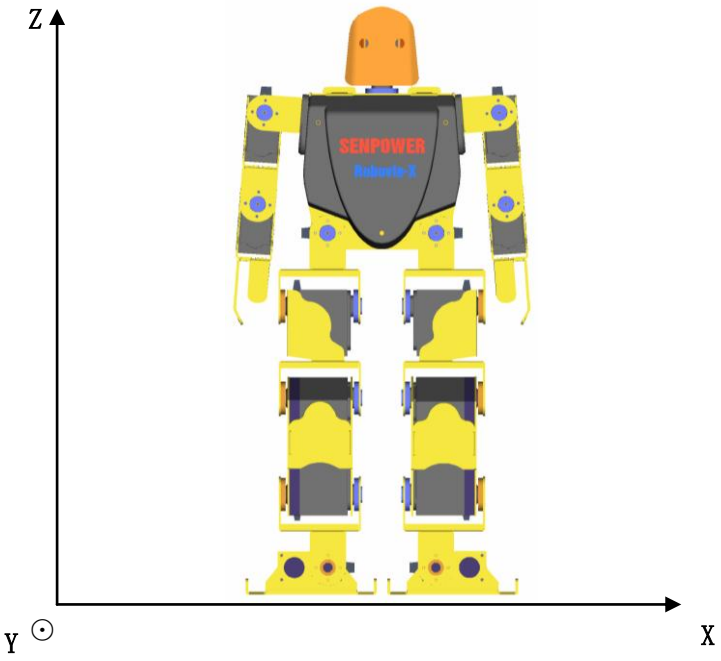
X 平面（侧平面）舵机



X 平面舵机：

position[0]---- PA0
position[1]---- PA1
position[2]---- PA2
position[3]---- PA3
position[4]---- PA4
position[5]---- PA5
position[6]---- PA6
position[7]---- PA7

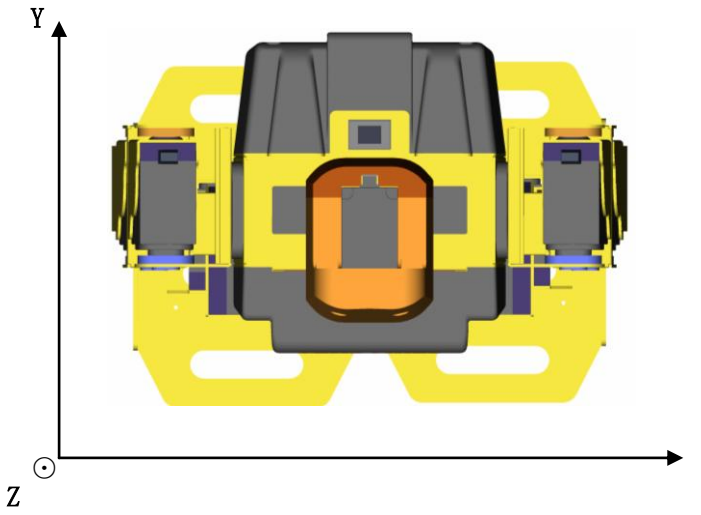
Y 平面（面平面）舵机



Y 平面舵机：

position[8] ---- PB0
position[9] ---- PB1
position[10]---- PB2
position[11]---- PB3
position[12]---- PB4
position[13]---- PB5
position[14]---- PB6
position[15]---- PB7

Z 平面（顶平面）舵机



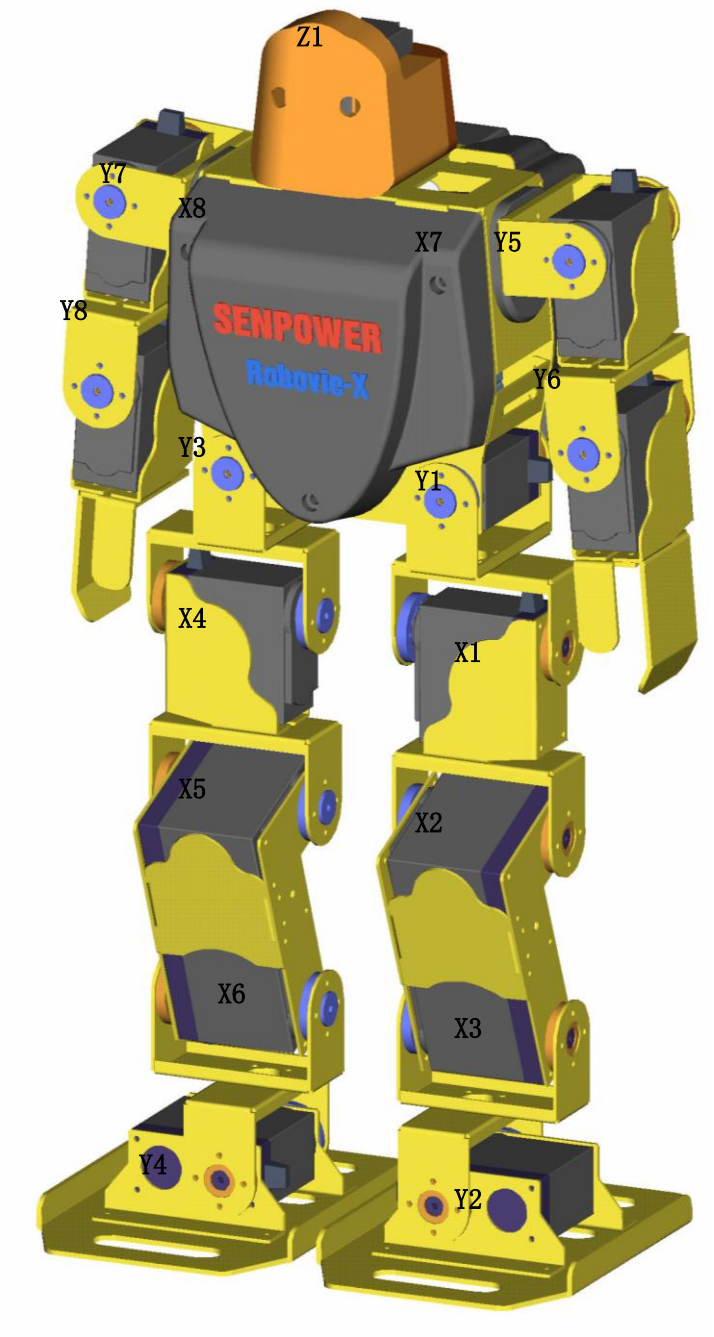
Z 平面舵机：

position[16]---- PC0
position[17]---- PC1
position[18]---- PC2
position[19]---- PC3
position[20]---- PC4
position[21]---- PC5
position[22]---- PC6
position[23]---- PC7

数组对应端口说明：

position[0]	——————>>	PA0	x1 舵机　x 轴
position[1]	——————>>	PA1	x2 舵机
position[2]	——————>>	PA2	x3 舵机
position[3]	——————>>	PA3	x4 舵机
position[4]	——————>>	PA4	x5 舵机
position[5]	——————>>	PA5	x6 舵机
position[6]	——————>>	PA6	x7 舵机
position[7]	——————>>	PA7	x8 舵机
position[8]	——————>>	PB0	y1 舵机　y 轴
position[9]	——————>>	PB1	y2 舵机
position[10]	——————>>	PB2	y3 舵机
position[11]	——————>>	PB3	y4 舵机
position[12]	——————>>	PB4	y5 舵机
position[13]	——————>>	PB5	y6 舵机
position[14]	——————>>	PB6	y7 舵机
position[15]	——————>>	PB7	y8 舵机
position[16]	——————>>	PC0	z1 舵机　z 轴
position[17]	——————>>	PC1	z2 舵机
position[18]	——————>>	PC2	z3 舵机
position[19]	——————>>	PC3	z4 舵机
position[20]	——————>>	PC4	z5 舵机
position[21]	——————>>	PC5	z6 舵机
position[22]	——————>>	PC6	z7 舵机
position[23]	——————>>	PC7	z8 舵机

2. 各个端口定义


Z1
Y7
X8
X7
Y5
Y8
SENPOWER
Robovie-X
Y6
Y3
Y1
X4
X1
X5
X2
X6
X3
Y4
Y2

八．多路比例尺插值法

在机器人的运动过程中，我们发现实际情况并不是完全按照我们的预想进行的。很多由于电子元件的精度而带来的误差，给整个机器人伺服系统造成严重的影响。误差一般分为 2 类，一类是时间误差，一类是位置误差。

时间误差：其中最为突出的是有 16MHZ 晶振的时间误差，当多台机器人进行千手观音表演时，由于每台机器人体内的晶振之间的误差，而且经过多次计算，计算的累计时间差也会很严重。机器人工作 5 分钟以上时，积累误差甚至达到 3 秒以上。

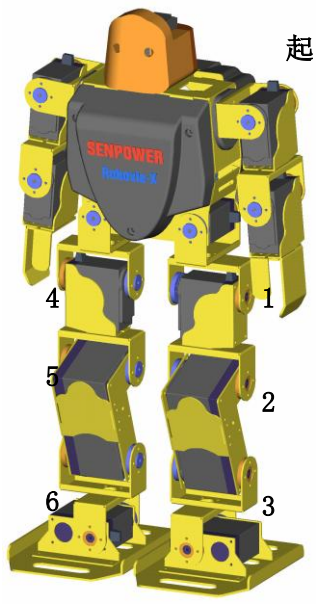
位置误差：其中最为突出的是舵机电位器线性度问题。当两个舵机给相同的 PWM 脉冲时，两个舵机的锁定角度不同，这种问题可以通过初始位置的设定解决，但是当两个舵机的 PWM 变化值相同时，居然两个舵机的角度变化值也不相同，甚至有的舵机误差达到 5%以上。这样的现象完全就是线性度不够造成的，对此我们有 2 种解决办法。

1． 采用高精度电位器；

2． 采用比例尺插值法；

如果仅采用第一种办法，尽管电位器的线性度可以做的很高，但是考虑到制造成本以及运动累计误差的影响，还是不能消除综合角度误差。那么我们就用比例尺插值法根治这现象。

1．以蹲起动作为例进行误差分析



机器人蹲起过程特点：①双腿着地；

②同起同落；

③脚、肩保持平行；

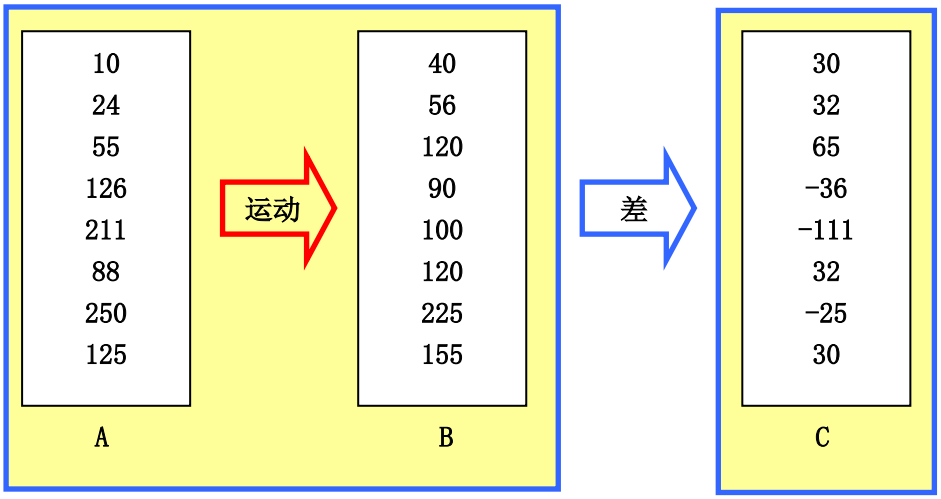
④胸平面轴不动；

根据这 4 个特点，我们采用积分程序将双腿的运动过程中的各个关节位置计算出来，并实时输送给舵机。共控制 6 个运动轴，分别为左腿的 X1、X2、X3 和右腿的 X4、X5、X6。

2. 多路并行插值控制法

如果机器人的运动是不规则运动或者说是为了向上面所说的局部关节误差而导致运动的变化量不成标准比例关系，那么下面讲述的这种方法就是一种必要的控制方法。

如下面的数据：有 8 个舵机参与运动，而每个舵机的运动角度是不同的，但是系统要求大家同时启动且同时停止，那么每个舵机的行程 C 不同。



观察行程 C 中的数据，有正负号，实际在程序编写过程中用标志位表示符号位，我们在运算过程中只取绝对值。取出绝对值后的 C 存储器内容如下图。

其中包括这样一组数：30、32、65、36、111、32、25、30。

这些数据中，最大的是 111，最小的是 25。几乎相差 4 倍之多。这种情况就用插值法根治问题。具体有 2 种插值方法：

（1） 最大值法----取这组数据中的最大值，也就是 111 了；

（2） 公共值法----取一个经验公共值，例如 100；

对各个舵机的 C 值做公共除法得到：

以 111 为分母：30/111、32/111、65/111、36/111、111/111、32/111、25/111、30/111。

优点：控制精度高，最大值为 111/111=1，其余的都小于 1；

缺点：做千手观音等动作时，由于各个机器人的舵机初始位置不同造成的不同步性；

以 100 为分母：30/100、32/100、65/100、36/100、111/100、32/100、25/100、30/100。

优点：同步性能好，适合做集体操时大量移植使用；

缺点：如果碰到 C 值较大时，控制精度有所下降；

最终要根据不同的环境选择使用不同的程序。

我们管这种方法俗称：P-TO-P 法；